spring新手攻略（9）-依赖注入(二)(Dependency Injection)

Posted Apr 4, 2014 | 评论

前言

在上篇文章中我们初步介绍了依赖注入（Dependency Injection）。今天将来介绍下spring框架中的两种依赖注入。

两种依赖注入（Dependency Injection）

- 基于构造函数的依赖注入
- 基于Setter方法的依赖注入

如果是强制性依赖的话建议使用基于构造函数的依赖注入，请看下面的例子：

```
1. package sirk_spring_tuto.demo;
2.
3. public class TV {
4. private Screen screen;
5.
6. public TV(Screen screen){
7.     this.screen = screen;
8. }
9.
10. public void powerOn(){
11.     this.screen.show();
12. }
13. }
14.
15.
16.
```

```
17. package sirk_spring_tuto.demo;
18.
19. public interface Screen {
20. public void show();
21. }
22.
23.
24. package sirk_spring_tuto.demo;
25.
26. public class CRTScreen implements Screen{
27.
28. public void show() {
29.     System.out.println("CRTScreen show beautiful girls");
30. }
31.
32. }
33.
34.
35.
36. <?xml version="1.0" encoding="UTF-8"?>
37. <beans xmlns="http://www.springframework.org/schema/beans"
38. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
39. xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.1.xsd">
40.
41.     <bean id="tv" class="sirk_spring_tuto.demo.TV">
42.      <constructor-arg ref="screen"></constructor-arg>
43.     </bean>
44.     <bean id="screen" class="sirk_spring_tuto.demo.CRTScreen"></bean>
45. </beans>
46.
47.
48.
49. package sirk_spring_tuto.demo;
50.
51. import org.springframework.context.ApplicationContext;
52. import org.springframework.context.support.ClassPathXmlApplicationContext;
53.
54. public class App {
55.
56. public static void main(String[] args) {
57.     ApplicationContext ctx = new ClassPathXmlApplicationContext("Beans.xml");
58.     TV tv = (TV) ctx.getBean("tv");
59.     tv.powerOn();
60.
61. }
62.
63. }
```

好了，通过上述程序运行，最终会打印CRTScreen show beautiful girls。代码很简单，首先定义了一个TV（电视机类），然后通过构造方法将Screen注入进去，这样电视机就有了显示器的 显示功能，但是电视机类并没有强耦合于Screen，内部的Screen属性只是一个接口，至于如何实现就看Spring了，我们在配置文件中将CTRScreen注入，于是就有了最终的结果。如果你不想要使用 CTRScreen了，想换个液晶显

示器，之需要修改配置文件就好了，这样能明白减少类之前的耦合是多么好了吧。同样的，以这个例子，我们再来看看另一种注入方式。

如果是可选的依赖注入，则建议使用基于Setter方式的依赖注入，如下：

```
1. package sirk_spring_tuto.demo;
2.
3. public class TV {
4. private Screen screen;
5.
6. /*public TV(Screen screen){
7.     this.screen = screen;
8. }*/
9.
10.
11.
12. public void powerOn(){
13.     this.screen.show();
14. }
15.
16. public Screen getScreen() {
17.     return screen;
18. }
19.
20. public void setScreen(Screen screen) {
21.     this.screen = screen;
22. }
23. }
24.
25.
26.
27. <?xml version="1.0" encoding="UTF-8"?>
28. <beans xmlns="http://www.springframework.org/schema/beans"
29. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
30. xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.1.xsd">
31.
32.     <!-- <bean id="tv" class="sirk_spring_tuto.demo.TV">
33. <constructor-arg ref="screen"></constructor-arg>
34.     </bean> -->
35.
36.     <bean id="tv" class="sirk_spring_tuto.demo.TV">
37.      <property name="screen" ref="screen"></property>
38.     </bean>
39.     <bean id="screen" class="sirk_spring_tuto.demo.CRTScreen"></bean>
40. </beans>
```

此处只贴出需要修改的两个文件，一个是TV类增加Setter方法，一个是beans配置文件的修改。

XML配置文件中使用p-namespace

当有多个Setter方法注入时，你可能会配置如下的文件：

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.1.xsd">

   <bean id="man" class="com.example.Man">
      <property name="name" value="sirk"/>
      <property name="wife" ref="lucy"/>
   </bean>

  <bean name="lucy" class="com.example.Woman">
      <property name="name" value="luck"/>
  </bean>
</beans>
```

如果使用p-namespace方式，则可以写成如下的方式：

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xmlns:p="http://www.springframework.org/schema/p"
     xsi:schemaLocation="http://www.springframework.org/schema/beans
     http://www.springframework.org/schema/beans/spring-beans-3.1.xsd">

   <bean id="man" class="com.example.Man"
       p:name="sirk"
       p:wife-ref="lucy"/>
   </bean>

   <bean name="lucy" class="com.example.Woman"
       p:name="luck"/>
   </bean>
</beans>
```

通过使用p-namespace 方式简化了对于引用和基本值的处理，之需要对引用bean的p加上-ref就好了。这样，spring会自动判断出该属性是一个引用。

小结

本文介绍了spring的两种依赖注入，下篇更精彩，敬请期待~

##文档信息

- 版权声明：自由转载-非商用-非衍生-保持署名 | Creative Commons BY-NC-ND 3.0